Thesis for the Master of Engineering

An Efficient and Effective Method to Find Uninteresting Items for Accurate Collaborative Filtering

Hyung-ook Kim

Graduate School of Hanyang University

August 2017

Thesis for Master of Engineering

An Efficient and Effective Method to Find Uninteresting Items for Accurate Collaborative Filtering

Thesis Supervisor: Sang-Wook Kim

A Thesis submitted to the graduate school of Hanyang University in partial fulfillment of the requirements for the degree of Master of Science

Hyung-ook Kim

August 2017

Department of Computer Science
Graduate School of Hanyang University

This thesis, written by Hyung-ook Kim,
has been approved as a thesis for the degree of Master of Engineering.

August 2017

Committee Chairman: Jauhyuk Cha (

Committee member: Sang-Wook Kim (Sang-Wook Kim

Committee member: Heejin Park (Park Heejin

Graduate School of Hanyang University

Table of Contents

List of Figures

List of Tables

ABSTRACT

An Efficient and Effective Method to Find Uninteresting Items for Accurate Collaborative Filtering

Hyung-ook Kim
Dept. of Computer and Software
Graduate School of
Hanyang University

Collaborative filtering methods suffer from a *data sparsity problem*, which indicates that the accuracy of recommendation decreases when the user-item matrix used in recommendation is sparse. To alleviate the data sparsity problem, researches on data imputation have been done. In particular, the *zero-injection* method, which finds uninteresting items and imputes zero values to those items for collaborative filtering, achieves significant improvement in terms of recommendation accuracy. However, the existing zero-injection method employs the One-Class Collaborative Filtering (OCCF) method that requires a lot of time. In this paper, we propose a fast method that finds uninteresting items rapidly with preserving high recommendation accuracy. Our experimental results show that our method is faster than the existing zero-injection method and also show that the recommendation accuracy using our method is slightly higher than or similar to that of the existing zero-injection method.

1. Introduction

1.1. Recommender Systems

As the number of items in online businesses grows exponentially, it becomes difficult for users to find the likable items. As a result, online businesses provide users with services that recommend items that users may like. As online businesses such as Ebay.com, Amazon.com, and Netflix.com equipped with *recommender systems technology* become successful, there has been a growing interest on recommender systems technology in the industry sector [Dia08].

The goals of recommender systems can be set differently as (1) rating prediction and (2) top-*N* recommendation [Hwa16]. In rating prediction, a recommender system predicts a *target user*'s ratings on her unrated items. Note that, the unrated item to be predicted is called a *target item*. In top-*N* recommendation, a recommender system selects *N* items to a target user among her unrated items in the order of her predicted preferences on them. Her preferences are predicted by the system. Generally, recommender systems used in real situations mostly aim at the top-*N* recommendation rather than prating prediction. In this paper, therefore, we focus our attention on top-*N* recommendation [Cre10, Ha12, Hwa16, Gun09].

Recommender systems can be classified into three categories: (1) *content-based methods*, (2) *collaborative filtering (CF) methods*, and (3) *hybrid methods* [Ado05]. To provide top-*N* recommendation to a target user, content-based methods build a user profile of the target user by referring to the content of her previously purchased (or rated highly) items, and recommend top-*N* items whose contents are most similar to those in the user profile [Lan95, Moo00, Paz07]. In the case of movies data, content-based methods employ movies' directors, actors, and genres as contents of the movies. On the other hand, CF methods predict a target user's ratings (or preferences) on unrated items by aggregating her similar users' ratings (or preferences), and provide top-N items with the highest predicted ratings to her [Bre98, Sar01, Zha05]. Figure 1 shows a process of CF methods. The CF methods are widely used in various applications because they do not depend on the contents of data and show the satisfactory accuracy in recommendation. Hybrid methods combine content-based methods and CF methods in various ways [Sch02, Mid04, Pop01, Cla99].

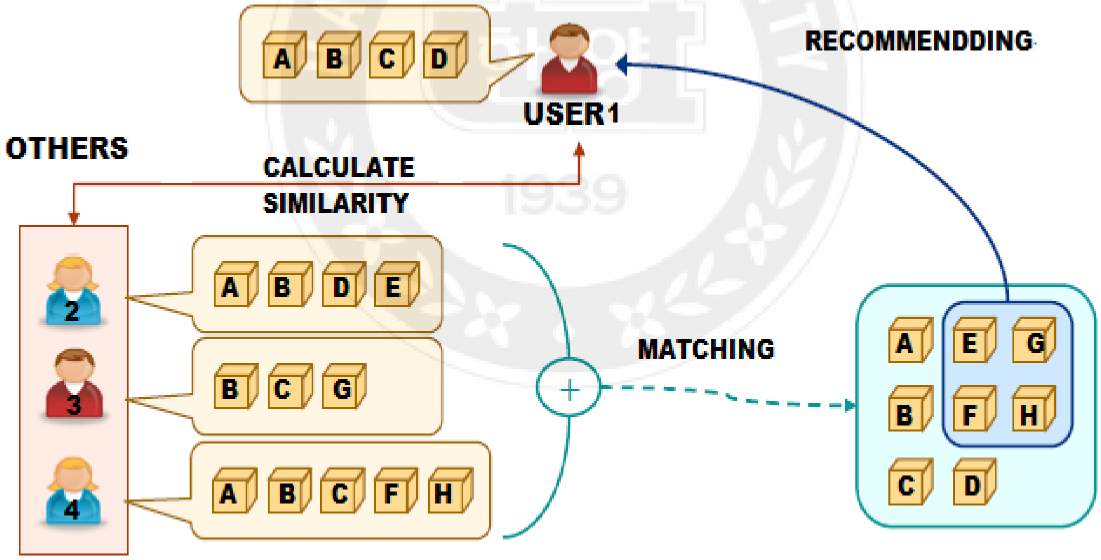


Figure 1: A process of collaborative filtering

CF methods, however, suffer from a *data sparsity problem*, which indicates that the accuracy of recommendation decreases when the user-item matrix used in CF is sparse [Ado05, Hwa16, Ma07, Ren12, Ren13, Sar01, Xue05]. The data sparsity indicates that the data has only a small fraction of rated items (i.e., *rated* user-item pairs) among a large number of total items (i.e., *all possible* user-item pairs). It is because most users rarely rate (or purchase) items. Ratings given by users to items can be represented as a *user-item matrix* whose row and column represent a user and an item, respectively, and the value of each cell in the matrix represents a rating score. Generally, the density of real-world data used in recommender systems is quite sparse, which indicates most cells in the user-item matrix are empty. For example, the density of MovieLens and Netflix data, widely used as benchmark data in recommendation researches, is only 4.25% and 1.17%, respectively [Sar01, Sin10]. Because CF methods figure out users' tastes and preferences only based on their ratings, the accuracy of recommendation becomes lower as the data gets sparser. To alleviate the data sparsity problem, two main categories of approaches have been studied: one is exploiting additional information from a trust network [Jam09, Ma08, Hwa13] or crowdsoucing [Erd14, Bac14, Cha16]. The other approach is data imputation [Hwa16, Ma07, Ren12, Ren13, Xue05].

1.2. Data Imputation

Recently, *data imputation* approaches have been proposed to alleviate the data sparsity problem. They first predict the ratings of unrated items and fill the empty cells in the user-item matrix with these predicted ratings [Hwa16, Ma07, Ren12, Ren13, Xue05], thereby increasing the density of the user-item matrix significantly. This makes us expect that the accuracy of recommendation is improved with these approaches because CF methods can figure out users' tastes more accurately with richer information in the user-item matrix. Also, previous research results via experiments showed that data imputation indeed improves the recommendation accuracy. In particular, Hwang et al. proposed a zero-injection method that finds an individual user's uninteresting items among unrated items and injects (or imputes) zero values to them, where a zero value means a user has no interest in the item [Hwa16]. Their method showed dramatic improvement in terms of the accuracy of top-*N* recommendation. We note, however, finding uninteresting items among unrated items is a very time-consuming process because it is based on the *One-Class Collaborative Filtering (OCCF)* method [Pan08], which is performed based on a *full-matrix factorization*.

1.3. Our Approach

In this paper, we propose a new method that finds uninteresting items more efficiently with preserving high recommendation accuracy. The proposed method selects a user who has smaller interest in items than other users and then selects an item (i.e. not interesting on item domain), among her unrated items, which is less appealing (i.e., popular) to all users than other items as an uninteresting item. In this way, the proposed method selects a user-item pair iteratively to find all the uninteresting items. Because the proposed method is much simpler than the existing OCCF method, it is much faster. Also, it provides the accuracy slightly higher than or similar to the OCCF method when used in top-*N* recommendation with zero-injection. In addition, it is easy to interpret and understand because it is quite intuitive.

For justifying the effectiveness and efficiency of the proposed method, we have performed extensive experiments. The results show that our method is 8 times faster than the existing OCCF method and that the recommendation accuracy with our method is comparable to that with the OCCF method.

1.4. Organization

The rest of this dissertation is organized as follows. In Section 2, we show why zero-injection benefits in CF based recommendation. In Section 3, we present our proposed method in detail. In Section 4, we evaluate our method in terms of the recommendation accuracy and performance via extensive experiments. In Section 5, we briefly review the related work to data imputation. In Section 6, we conclude the paper and discuss the future work.

2. Benefit from Zero-Injection

2.1. Background and Issues

The user-based CF and the item-based CF are representative methods of collaborative filtering [Ado05, Bre98, Sar01]. They find a group of users whose preferences are similar to that of the target user (in user-based CF) and a group of items whose popularity is similar to that of the target item (in item-based CF)[1]. It is important to find similar users whose preferences are similar to that of the target user, so accurate similarity computation is needed.

In case the user-item matrix is sparse in CF, the similarity between users could be inaccurately computed, which makes the recommendation accuracy unsatisfactory [Hwa16]. The Pearson correlation coefficient (PCC) and the cosine-based similarity are widely used as measures to compute the similarity between a pair of users [Sar01]. Because they consider only those items *commonly rated* by the two users, if there are only a few of such items between them, their similarity might be inaccurate. The similarity between two users who have a few common items can be higher than the similarity between two users who have a lot of common items.

For example, we assume that there are three users, A, B, and C as in Table 1. Each column corresponds to a movie and each row does a user. Each number in each cell (i, j) represents a rating score on an item j given by a user i. Users A and C have watched comedy movies while user B has watched horror movies. In this case, we can conjecture

[1] Without loss of generality, we explain the benefit by zero-injection only in the aspect of the user-based CF here. The explanation, however, holds with the item-based CF.

Table 1: A user-item matrix for user A, B, and C.

User	Comedy movies									Horror movies								
A	3	2	5	3	2	1	1	5	4	3	4							
B								4	3	2	1	1	2	3	2	1	3	3
C	2	1	2	3	3	2	2	3	4	2	1							

that the similarity between users A and C is higher than that between users A and B. The PCC, however, between users A and B is computed as 0.632, and that between users A and C is computed as 0.235. We know that this result is counter-intuitive.

2.2. Problem Solving with Zero-Injection

We could solve this problem if we have users' preferences on unrated items. Previous researches addressed *data imputation* that predicts users' preferences on unrated items and fill the unrated items with the predicted scores [Ma07, Ren12, Ren13, Xue05]. It is unavoidable, however, to have errors in this prediction. In Table 1, we assume that all the unrated items are uninteresting items. When we apply zero-injection to those uninteresting items, the PCC between users A and B is computed as -0.088, and that between users A and C is computed as 0.764. This result coincides with our intuitions, unlike the previous result. Although all the unrated items are assumed as uninteresting items in this example, it is lucky to have a reasonable result. If we identify uninteresting items more accurately among unrated items and inject zeros on them in the user-item matrix, it helps to compute the similarity more accurately, which could lead to more accurate recommendation by CF. Therefore, in this paper, we address the method to find uninteresting items accurately as well as fastly to perform zero-injection to uninteresting items for accurate recommendation.

3. Proposed Method

3.1. Overview

As mentioned earlier, the zero-injection method proposed by Hwang et al. [Hwa16] has a bottleneck in finding uninteresting items. This is because it employs the OCCF method that is time-consuming since it is performed based on a full-matrix factorization. In this paper, we propose a simple yet effective method to find uninteresting items among unrated items. Specifically, we first select a user who is less interested in items than other users, and then select an item, among her unrated items, which is less appealing (i.e., popular) to all users than other items as an uninteresting item. To find uninteresting items, we perform this selection method iteratively until a predefined number of uninteresting items are obtained. Then, we perform zero-injection for uninteresting items in the user-item matrix for providing accurate top-*N* recommendation by CF.

3.2. Selection Method for Uninteresting Items

First, we select a user having a few rated items as a user who is less interested in items because she is not that interested in the items' domain. Subsequently, she will not be interested in most of unrated items as well. For example, in the case of movies recommendation, a user having a few ratings might not like to watch movies, so it is highly unlikely that she is interested in most of other her unrated (i.e., not watched) movies.

Second, among her unrated items, we select an item rated by a number of users (i.e., popular) as the item that is less appealing to her. This kind of items are popular and could mostly have good quality. For this popular item, although it is unrated, the user might be aware of its existence. Thus, we can say that she has not given a rating on the item because she is not interested in it. For example, in movies recommendation, if a user has not watched a very popular movie such as “Titanic” or “Starwars”, she is likely to be aware of it already but is not interested in it.

In finding uninteresting items, if we simply use *a number* in order to select a user who rated a small number of items and an item rated by a large number of users, there might be a problem that *the same user or item will be selected repeatedly*. This is because the number of users (in selecting items) and the number of items (in selecting users) in the user-item matrix will not be changed during the process of selecting users and items.

3.2.1. User Selection

To solve this problem of selecting identical users and items repetitively, we propose a method that selects a user and an item by the *probability* rather than by the number. We assign a probability to every user that is proportional to the number of her unrated items, then select a user based on the assigned probability. The probability assigned to each user is computed as follows:

$$P(u_i) = \frac{|I| - |I(i)|}{\sum_{u_k \in U}(|I| - |I(k)|)} \tag{1}$$

where $P(u_i)$ is the probability of selecting a user i, $I(i)$ is a set of rated items by user i, and U is a set of all users, I is a set of all items. In our user selection process, a user i is selected if a random number n_{rand} generated within [0,1] satisfies the following formula.

$$\sum_{k=1}^{i-1} P(u_k) \leq n_{rand} < \sum_{k=1}^{i} P(u_k) \tag{2}$$

3.2.2. Item Selection

After selecting the user, we select an item which is rated by a number of other users. Similar to the user selection, we assign the probability to her unrated item that is proportional to the number of ratings of the item given by other users, then we select an item based on the assigned probability. The probability assigned to each item is computed as follows:

$$P(i_j) = \frac{|U(j)|}{\sum_{i_k \in I(s)} |U(k)|} \tag{3}$$

where $P(i_j)$ is the probability of selecting item j, $U(j)$ is a set of users who rated an item j, and $I(s)$ is a set of rated items by selected user s. In our item selection process, an item j is selected if a random number n_{rand} generated within [0,1] satisfies the following formula.

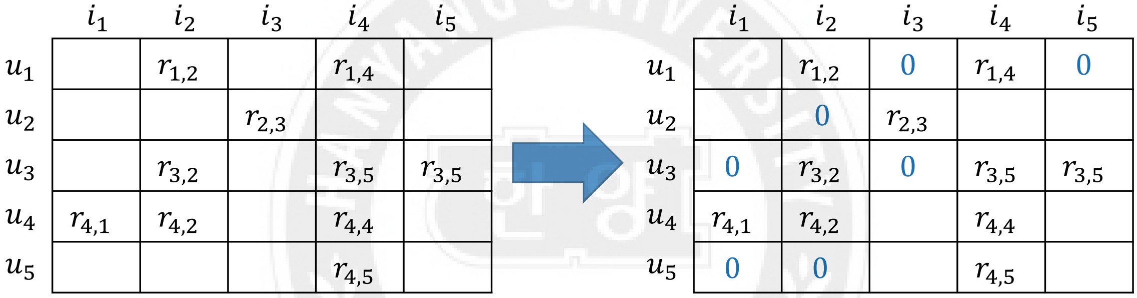
$$\sum_{k=1}^{j-1} P(i_k) \leq n_{rand} < \sum_{k=1}^{j} P(i_k) \tag{4}$$

3.3. Zero-Injection

The selection of an uninteresting item explained above is iteratively performed until a predefined number of uninteresting items are obtained. Then, those items are injected with zero values. It is important to determine the number of actual uninteresting items. This is because, if we inject a zero to an interesting item of a user, her preference on the item could be distorted, which makes the recommendation accuracy decreased. In this paper, we set the portion of uninteresting items to be filled by zero-injection as a parameter, called θ. The optimal parameter value for θ depends on data sets. In this paper, in order to find the optimal θ in our data set, we perform experiments to examine the change of recommendation accuracy according to θ.

3.4. Recommendation Method

After zero-injection, we have a much denser user-item matrix where the data sparsity problem is alleviated as in Figure 2. Then, we predict a user i's preference on an item j through the user-based CF as follows:



	i_1	i_2	i_3	i_4	i_5
u_1		$r_{1,2}$		$r_{1,4}$	
u_2			$r_{2,3}$		
u_3		$r_{3,2}$		$r_{3,5}$	$r_{3,5}$
u_4	$r_{4,1}$	$r_{4,2}$		$r_{4,4}$	
u_5				$r_{4,5}$	

	i_1	i_2	i_3	i_4	i_5
u_1		$r_{1,2}$	0	$r_{1,4}$	0
u_2		0	$r_{2,3}$		
u_3	0	$r_{3,2}$	0	$r_{3,5}$	$r_{3,5}$
u_4	$r_{4,1}$	$r_{4,2}$		$r_{4,4}$	
u_5	0	0		$r_{4,5}$	

Figure 2: A user-item matrix and a zero-injected matrix

$$r_{i,j} = \overline{u_i} + \frac{\sum_{u_k \in U} sim(u_i, u_k) \cdot (r_{k,j} - \overline{u_k})}{\sum_{u_k \in U} sim(u_i, u_k)} \tag{5}$$

where $\overline{u}_i$ is the average rating on items by user i and $sim(u_i, u_k)$ is the similarity between users i and k computed by the PCC computed as follows:

$$sim(u_i, u_j) = \frac{\sum_{i_k \in I(i) \cap I(j)} (r_{i,k} - \overline{u_i}) \cdot (r_{i,k} - \overline{u_i})}{\sqrt{\sum_{i_k \in I(i) \cap I(j)} (r_{i,k} - \overline{u_i})^2} \cdot \sqrt{\sum_{i_k \in I(i) \cap I(j)} (r_{j,k} - \overline{u_j})^2}} \tag{6}$$

After predicting the target user's preference on her unrated items, we recommend top-*N* unrated items in the order of her predicted preferences. If we want to provide top-*N* recommendation to multiple users, we perform the same process for different users on the same injected matrix (i.e., predicting the user's preferences on her unrated items and recommending her top-*N* items).

4. Evaluation

4.1. Experimental Setup

In order to evaluate the performance and accuracy of the proposed method, we conducted extensive experiments with the MovieLens 100K data set [Res94], which is widely used to evaluate recommendation systems [Ha12, Hwa16, Ma07, Ren12, Ren13, Sar01, Xue05]. It contains 943 users having at least 20 ratings and 1,682 movies. The number of ratings, given as an integer between 1 and 5, is 100,000. We divide the data set into a training set and a test set with proportion of 4 to 1 and perform a 5-fold cross validation on them. We employ P@*N*, R@*N*, and F1@*N* as evaluation metrics[2], where *N* indicates the number of items recommended.

$$Precision = \frac{|\{correct\ items\}| \cap |\{recommended\ items\}|}{|\{recommended\ items\}|} \tag{7}$$

$$Recall = \frac{|\{correct\ items\}| \cap |\{recommended\ items\}|}{|\{correct\ items\}|} \tag{8}$$

$$F1\ \ score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{9}$$

4.2. Experimental Results

In Experiment 1, we analyze the change of recommendation accuracy with different parameters θ from 10% to 100%. Table 2 shows the result where bold-faced ones indicate the best accuracies in each metric. The proposed method shows the highest accuracy when

[2] P@*N*, R@*N*, and F1@*N* indicates precision@*N*, recall@*N*, and F1 score@*N*, respectively.

Table 2: Recommendation accuracy of our method according to parameter θ.

θ	P@5	P@10	P@15	R@5	R@10	R@15	F1@5	F1@10	F1@15
10%	0.0982	0.0942	0.0876	0.1046	0.1878	0.2528	0.1013	0.1255	0.1301
20%	0.1924	0.1558	0.1312	0.1974	0.2992	0.3564	0.1949	0.2049	0.1918
30%	0.2102	0.1632	0.1376	0.2162	0.3144	0.3768	0.2132	0.2149	0.2016
40%	**0.2110**	**0.1652**	0.1390	**0.2190**	**0.3182**	0.3820	**0.2149**	**0.2175**	0.2038
50%	0.2100	0.1638	**0.1392**	0.2184	0.3144	**0.3870**	0.2141	0.2154	**0.2048**
60%	0.2098	0.1628	0.1380	0.2186	0.3120	0.3830	0.2141	0.2140	0.2029
70%	0.2094	0.1628	0.1382	0.2166	0.3152	0.3846	0.2129	0.2147	0.2033
80%	0.2070	0.1630	0.1368	0.2170	0.3148	0.3810	0.2119	0.2148	0.2013
90%	0.2074	0.1618	0.1370	0.2176	0.3134	0.3816	0.2124	0.2134	0.2016
100%	0.2088	0.1622	0.1372	0.2174	0.3136	0.3826	0.2130	0.2138	0.2020

θ is in the range of 40-50%. This implies that 40-50% of unrated items in the MovieLens data set might be uninteresting items. When θ is higher than 50%, the accuracy gradually decreases with some fluctuations. In case θ is larger than 50%, the items interesting to users could be incorrectly regarded as uninteresting. Therefore, the zero-injection to them (i.e., interesting items) causes recommendation accuracy decreased. In the next experiments, we set θ as 40%, which provides the best accuracy.

In Experiment 2, we examine the recommendation accuracies with *four variants* of our method for finding uninteresting items: (M1: ours) selecting a user having *a few* items and an item rated by *a number of* users, (M2) a user having *a few* items and an item rated by *a few* users, (M3) selecting a user having *a number of* items and an item rated by *a few* users, and (M4) a user having *a number of* items and an item rated by *a number of* users. In this experiment, we find uninteresting items by each of the four variants and compare the accuracies of recommendation obtained by it.

Figure 3 shows the result. The *x*-axis represents the number of items to be recommended, *N*, and the *y*-axis does P@*N*, R@*N*, and F1@*N* in Figure 3(a), 3(b), and 3(c), respectively. CF equipped with our method universally outperforms that with the other three variants in all *N* and all metrics. This result indicates (1) a user having a few rated items is not interested in the items' domain, and (2) her item rated by a number of users, in particular, is less appealing to her.

In Experiment 3, we evaluate CF equipped with our method in comparison with other CF methods based on data imputation. We select the AdaM[3] by Ren et al. [Ren13], the most accurate CF method employing the data imputation with predicted ratings, and the zero-injection method[4] by Hwang et al. [Hwa16] for comparisons with our method. In this experiment, parameter θ of our method is set as 40%, the parameter of AdaM, λ, is set as 0.4, and the parameters in Hwang's method, λ, #iterations, and #factors are set as 0.015, 30, and 20, respectively.

Table 3 shows the result in terms of P@N, R@N, and F1@N for each method. The recommendation accuracy of zero-injection methods (Hwang's and ours) is shown much higher than that of the AdaM. In particular, the accuracy with our method is 4.6 times higher than that of the AdaM in F1@5. Compared with Hwang's method, our method does not show a big difference in accuracy, but is a bit more accurate.

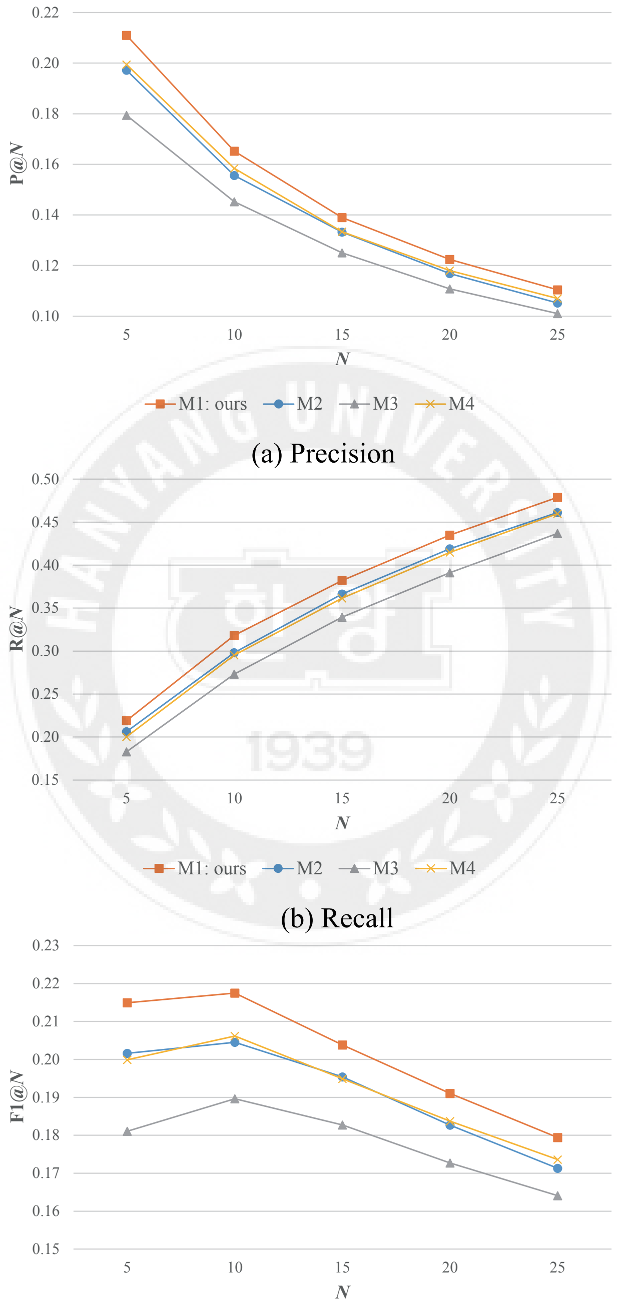
Table 3: Recommendation accuracies of our and existing methods.

	P@5	P@10	P@15	R@5	R@10	R@15	F1@5	F1@10	F1@15
Ours	**0.2110**	**0.1652**	0.1390	**0.2190**	**0.3182**	0.3820	**0.2149**	**0.2175**	0.2038
AdaM	0.0510	0.0460	0.0440	0.0430	0.0760	0.1080	0.0467	0.0573	0.0625
Hwang's	0.2064	0.1644	**0.1402**	0.2104	0.3156	**0.3868**	0.2084	0.2162	**0.2058**

[3] It is known that the recommendation accuracy of the AdaM is higher than that of the EMDP [Ma07] and the AutAI [Ren12], so we only selected AdaM for the comparison.

[4] We called the zero-injection method that employs the OCCF method Hwang's method.

In Experiment 4, we compare the execution times of our method and the OCCF method in Hwang's method in finding uninteresting items. We ran our experiments on a PC equipped with an Intel Core-i5 3.4GHz and 16GB RAM running the 64bit Windows 7 operating system. We set all the parameters for each method exactly same as in Experiment 3. The experimental result shows that the execution time of our method is 63.67 seconds and that of Hwang's method is 511.03 seconds, which indicates our method performs about 8 times faster than Hwang's method. In summary, our method has an advantage of fast execution while showing recommendation accuracy comparable to the existing one.



(a) Precision

(b) Recall

(c) F1 score

Figure 3: Recommendation accuracies with different methods of finding uninteresting items.

5. Related Work

There are various methods of data imputation to alleviate the data sparsity problem in CF methods. In this section, we briefly review them.

Xue et al. proposed a data impression method, SCBPCC, that uses clustering method to solve data sparsity problem and scalability [Xue05]. The SCBPCC performs k-means algorithm on the user, and performs data imputation for each user by using the ratings of users who belong to the user's cluster. When predicting the target user's rating, the similarity between the target user and the centroid of the clusters is measured to select the closest clusters. For the users in the selected clusters, we select neighbors by calculating the similarity with the target user, and aggregate the ratings of neighbors to predict the ratings. SCBPCC views imputed rating and original rating differently, so it assigns different weights between imputed rating and original rating at the time of similarity calculation and aggregation.

Ma et al. proposed the EMDP to impute empty cells in the user-item matrix with trustable scores [Ma07]. The EMDP predicts the scores for empty cells by using existing CF methods. When predicting an empty cell (or item), if it cannot find enough users (or items) similar to the user corresponding to the empty cell, it does give up the imputation on the cell. Through a series of experiments, they showed that their method achieves recommendation accuracy higher than other existing methods.

Ren et al. proposed two methods of data imputation, called AutAI and AdaM [Ren12, Ren13]. They used the concept of a *key set*, which is a set of cells in a user-item matrix that are important to predict a rating of a target user on a target item by CF. The AutAI and AdaM impute the empty cells in a key set by CF for more accurate recommendation.

Ren et al. proved that their methods should be more accurate than existing methods in a theoretical perspective, and also showed that their methods are effective through a series of experiments. Because the AdaM selects more cells as a key set than the AutAI, it predicts the target user's rating on the target item with more imputed ratings than the AutAI. They experimentally showed that the AdaM is more accurate than the AutAI.

Hwang et al. defined a new notion of uninteresting items and also proposed a method to identify those items and to use them for more accurate CF [Hwa16]. They defined an uninteresting item as an item unrated by a user because she knew but did not like it. To find uninteresting items, they used the OCCF method [Pan08]. Also, they proposed the zero-injection method that imputes a zero value into uninteresting items in a user-item matrix. Imputing a zero value to an item for a user indicates that she does not like the item. Their method showed a dramatic improvement in the accuracy of top-*N* recommendation. As mentioned in Section 1, the OCCF method, used to identify uninteresting items, is quite time-consuming because it is based on full-matrix factorization.

6. Conclusions

In this paper, we have addressed how to find uninteresting items efficiently and effectively for zero-injection used to alleviate the data sparsity problem. To the end, we have proposed a simple but effective method to exploit a user having a few rated items and an item rated by a number of users. For justifying the effectiveness and efficiency of the proposed method, we have performed extensive experiments. The results show that our method is 8 times faster than the existing OCCF method and that the recommendation accuracy with our method is comparable to that with the OCCF method.

As further study, we plan to do research on how to use zero-injection and data imputation together in CF. We expect that this research direction will give important insights towards solving the data sparsity problem, thereby achieving a significant improvement in terms of recommendation accuracy.

References

[Ado05] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering, TKDE*, Vol. 17, No. 6, pages 734–749, 2005.

[Bac14] Y. Bachrach, S. Ceppi, I. A. Kash, P. Key, F. Radlinski, E. Porat, M. Armstring, and V. Sharma. Building a personalized tourist attraction recommender system using crowdsourcing. In *Proceedings of ACM International Conference on Autonomous Agents and Multi-Agent Systems, AAMAS*, pages 1631–1632, 2014.

[Bre98] J. S. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of Conference on Uncertainty in Artificial Intelligence, UAI*, pages 43–52, 1998.

[Cha16] S. Chang, F. M. Harper, L. He, and l. G. Terveen. CrowdLens: Experimenting with crowd-powered recommendation and explanation. In *Proceedings of International AAAI Conference on Web and Social Media*, pages 52–61, 2016.

[Cla99] M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, and M. Sartin. Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR Workshop on Recommendation Systems*, Vol. 60, 1999.

[Cre10] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of ACM Conference on Recommender Systems, RecSys*, pages 39–46, 2010.

[Dia08] M. B. Dias, D. Locher, M. Li, W. El-Deredy, and P. J.G. Lisboa. The value of personalised recommender systems to e-business: A case study. In *Proceedings of ACM Conference on Recommender Systems, RecSys*, pages 291–294, 2008.

[Erd14] M. Erdt and C. Rensing. Evaluating recommender algorithms for learning using crowdsourcing. In *Proceedings of IEEE International Conference on Advanced Learning Technologies, ICALT*, pages 513–517, 2014.

[Gan11] Z. Gantner et al. Mymedialite: A free recommender system library. In *Proceedings of ACM Conference on Recommender Systems, RecSys*, pages 305–308, 2011.

[Gun09] A. Gunawardana and G. Shani. A survey of accuracy evaluation metrics of recommendation tasks. *Journal of Machine Learning Research, JMLR*, Vol. 10, pages 2935–2962, 2009.

[Ha12] J. Ha, S. Kwon, S. Kim, C. Faloutsos, and S. Park. Top-n recommendation through belief propagation. In *Proceedings of ACM International Conference on Information and Knowledge Management, CIKM*, pages 2343–2346, 2012.

[Hwa13] W. Hwang, S. Li, S. Kim, and H. J. Choi. Exploiting trustors as well as trustees in trust-based recommendation. In *Proceedings of IEEE International Conference on Information and Knowledge Management, CIKM*, pages 1893–1896, 2013.

[Hwa16] W. Hwang, J. Parc, S. Kim, J. Lee, and D. Lee. “Told you I didn’t like it”: Exploiting uninteresting items for effective collaborative filtering. In *Proceedings of IEEE International Conference on Data Engineering, ICDE*, pages 349–360, 2016.

[Jam09] M. Jamali and M. Ester. TrustWalker: A random walk model for combining trust-based and item-based recommendation. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining, KDD*, pages 397–406, 2009.

[Lan95] K. Lang. Newsweeder: Learning to filter netnews. In *Proceedings of International Conference on Machine Learning*, pages 331–339, 1995.

[Ma07] H. Ma, I. King, and M. R. Lyu. Effective missing data prediction for collaborative filtering. In *Proceedings of ACM International Conference on Research and Development in Information Retrieval, SIGIR*, pages 39–46, 2007.

[Ma08] H. Ma, H. Yang, M. R. Lyu., and I. King. SoRec: Social recommendation using probabilistic matrix factorization. In *Proceedings of ACM International Conference on Information and Knowledge Management, CIKM*, pages 931–940, 2008.

[Mid04] S. E. Middleton, N. R. Shadbolt, and D. C. D. Roure. Ontological user profiling in recommender systems. *ACM Transactions on Information Systems, TOIS*, Vol. 22, No. 1, pages 54–88, 2004.

[Moo00] R. J. Mooney and L. Roy. Content-based book recommending using learning for text categorization. In *Proceedings of ACM Conference on Digital Libraries*, pages 195–204, 2000.

[Pan08] R. Pan et al. One-class collaborative filtering. In *Proceedings of IEEE International Conference on Data Mining, ICDM*, pages 502–511, 2008.

[Paz07] M. J. Pazzani and D. Billsus. Content-based recommendation systems. *The adaptive web*, pagse 325-341, 2007.

[Pop01] A. Popescul, D. M. Pennock, and S. Lawrence. Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of Conference on Uncertainty in Artificial Intelligence, UAI*, pages 437–444, 2001.

[Ren12] Y. Ren et al. The efficient imputation method for neighborhood-based collaborative filtering. In *Proceedings of ACM International Conference on Information and Knowledge Management, CIKM*, pages 684–693, 2012.

[Ren13] Y. Ren et al. AdaM: Adaptive-maximum imputation for neighborhood-based collaborative filtering. In *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM*, pages 628–635, 2013.

[Res94] P. Resnick et al. Grouplens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM Conference on Computer Supported Cooperative Work, CSCW*, pages 175–186, 1994.

[Sar01] B. Sarwar et al. Item-based collaboration filtering recommendation algorithms. In *Proceedings of International Conference on World Wide Web, WWW*, pages 285–295, 2001.

[Sch02] A. I. Schein, A. Popescul, L. H. Ungar, D. M. Pennock. Methods and metrics for cold-start recommendations. In *Proceedings of ACM International Conference on Research and Development in Information Retrieval, SIGIR*, pages 253–260, 2002.

[Sin10] V. Sindhwani et al. One-class matrix completion with low-density factorizations. In *Proceedings of IEEE International Conference on Data Mining, ICDM*, pages 1055–1060, 2010.

[Xue05] G. -R. Xue et al. Scalable collaborative filtering using cluster-based smoothing. In *Proceedings of ACM International Conference on Research and Development in Information Retrieval, SIGIR*, pages 114–121, 2005.

[Zha05] S. Zhang et al. Using singular value decomposition approximation for collaborative filtering. In *Proceedings of IEEE International Conference on E-Commerce Technology, CEC*, pages 257–264, 2005.

국 문 요 지

Ebay.com, Amazon.com 등과 같은 많은 온라인 쇼핑몰에서 추천 시스템이 성공적으로 널리 이용됨에 따라, 추천 시스템에 대한 중요성이 증가하고 있다. 추천 시스템에 널리 이용되는 방법 중 하나는 협업 필터링 (collaborative filtering) 방법이다. 협업 필터링 방법은 유저들이 아이템에 부여한 평점들을 분석하여 타겟 유저와 취향이 유사한 유저들인 이웃 유저를 찾고, 이웃 유저들의 평점을 종합하여 타겟 유저가 평가하지 않은 아이템들에 대한 평점을 예측한다. 최종적으로 평점이 높게 예측된 아이템들을 타겟 유저에게 추천해준다. 협업 필터링 방법은 아이템의 내용 정보 (contents)에 구애 받지 않으며, 만족할만한 추천 정확도를 보여준다는 장점으로 널리 연구되고 사용되어 왔다.

그러나 협업 필터링 방법은 추천에 사용되는 유저-아이템 평점 행렬이 희소할 때 추천 정확도가 감소하는 문제인 데이터 희소성 (data sparsity) 문제가 있다. 데이터 희소성 문제를 해결하기 위해서, 데이터 임퓨테이션 (data imputation) 연구가 수행되었다. 특히, 사용자의 무관심 아이템을 찾아서 해당 아이템들에 평점 0점을 채워 넣는 방법인 zero-injection 방법은 추천 정확도를 굉장히 크게 향상시켰다. 하지만 기존 zero-injection 방법은 One-Class Collaborative Filtering (OCCF) 방법을 이용하여 무관심 아이템을 찾는데, 이 방법은 많은 계산 시간을 필요로 한다. 본 논문에서는 zero-injection의 높은 추천 정확도를 유지하면서, 무관심 아이템을 빠르게 찾을 수 있는 새로운 zero-injection 방법을 제안한다. 제안하는 방법은 무관심 아이템을 (1) 다른 유저에 비해 아이템들에 관심이 적은 유저와 (2) 다른 아이템들에 비해 인기가 없을 아이템을 찾는 방법을 통해 찾는다. 제안하는 방법은 기존 방법에서 활용하는 OCCF에 방법에 비해 알고리즘이 간단하기 때문에 수행 시간이 짧다. 실험 결과를 통해서 제안하는 방법이 기존 zero-injection 방법보다 수행 시간이 적다는 것을 보였고, 또한 추천 정확도 역시 기존 zero-injection 방법보다 조금 높거나 유사한 결과를 보였다.

감사의 글

데이터 및 지식공학 연구실에 들어와서 석사 학위를 받기까지 짧지 않은 시간이 걸렸습니다. 석사과정으로 들어와서 석박통합과정으로 전환하고 다시 석사과정으로 전환하는 우여곡절이 많았지만, 제게 다 의미 있고 값진 시간이었습니다.

제가 우여곡절 속에서도 무사히 졸업할 수 있었던 것은 많은 분의 도움이 있었기 때문입니다. 먼저, 연구와 논문 작성에서 많은 지도를 해주신 김상욱 교수님께 진심으로 감사드립니다. 연구가 잘 안 되고 힘들 때, 많은 격려를 해주셔서 어려운 상황들을 이겨내고 지금까지 연구할 수 있었습니다. 제 학위 논문 심사를 위해서 바쁜 일정 중에도 귀중한 시간을 내주어주신 차재혁 교수님과 박희진 교수님께도 감사의 말씀을 드립니다.

그리고 3년 반이라는 짧지 않은 시간 동안 같이 생활한 연구실 선배님과 동기, 후배들에게도 감사의 말씀 드립니다. 제가 입학한 이후로 사형으로서 제 연구를 이끌어 주신 지운 선배를 비롯하여 원석 선배, 현교 선배, 지원 선배, 용연이형, 동규, 윤석이형에게 감사를 전합니다. 꽤 오랜 시간 기숙사 룸메이트로 지내면서 연구 외적으로도 많은 도움 받은 윤용이형에게도 고마움을 전합니다. 석사 초기에 졸업하여 오랜 시간 함께 하지 못했지만 편하게 대해준 성우, 병문 선배, 주안이형, 창욱이형에게도 감사를 전합니다. 동기로 같이 입학하여 연구실 생활을 즐겁게 보낼 수 있게 해준 태희형, 규환, 특히 논문 작성에 많은 도움을 준 연창에게 감사를 전합니다. 동기는 아니었지만, 동기 같은 헬스 코치님 준호형에게도 감사를 전합니다. 연구에 많은 도움을 준 영남을 비롯하여 희정, 재근, 유진, 나영에게도 고마움을 전합니다. 반년간 룸메이트였던 동균이를 비롯하여 장완, 상현, 경재, 명환에게도 고마움을 전합니다. 짧은 기간이었지만 같이 연구한 홍균이와 신입생인 동규, 석진, 태리, 진수, 혜경에게도 감사를 전합니다.

마지막으로 힘들 때 짜증 내도 다 받아주고 지금의 내가 있게 해준 엄마, 아빠 그리고 형주에게 진심으로 고마움을 전합니다. 좀 까칠한 아들이자 형이지만 이해해줘서 감사합니다. 그리고 항상 응원해주신 이모들과 하늘에 계신 외할머니께 진심으로 감사드립니다.

일일이 열거하지는 못하였지만, 물심양면으로 도와주신 많은 분께 진심으로 감사드립니다.

연구 윤리 서약서

본인은 한양대학교 대학원생으로서 이 학위논문 작성 과정에서 다음과 같이 연구 윤리의 기본 원칙을 준수하였음을 서약합니다.

첫째, 지도교수의 지도를 받아 정직하고 엄정한 연구를 수행하여 학위논문을 작성한다.

둘째, 논문 작성시 위조, 변조, 표절 등 학문적 진실성을 훼손하는 어떤 연구 부정행위도 하지 않는다.

셋째, 논문 작성시 논문유사도 검증시스템 "카피킬러"등을 거쳐야 한다.

2017년06월13일

학위명 : 석사

학과 : 컴퓨터·소프트웨어학과

지도교수 : 김상욱

성명 : 김형욱 (서명)

한양대학교대학원장귀하

Declaration of Ethical Conduct in Research

I, as a graduate student of Hanyang University, hereby declare that I have abided by the following Code of Research Ethics while writing this dissertation thesis, during my degree program.

"First, I have strived to be honest in my conduct, to produce valid and reliable research conforming with the guidance of my thesis supervisor, and I affirm that my thesis contains honest, fair and reasonable conclusions based on my own careful research under the guidance of my thesis supervisor.

Second, I have not committed any acts that may discredit or damage the credibility of my research. These include, but are not limited to : falsification, distortion of research findings or plagiarism.

Third, I need to go through with Copykiller Program(Internet-based Plagiarism-prevention service) before submitting a thesis."

JUNE 13, 2017

Degree : Master

Department : DEPARTMENT OF COMPUTER SCIENCE

Thesis Supervisor : Sang-Wook Kim

Name : KIM HYUNG OOK (Signature)